PRV
PATENT- OCH REGISTRERINGSVERKET
Patentavdelningen

Intyg
Certificate

Härmed intygas att bifogade kopior överensstämmer med de handlingar som ursprungligen ingivits till Patent- och registreringsverket i nedannämnda ansökan.

This is to certify that the annexed is a true copy of the documents as originally filed with the Patent- and Registration Office in connection with the following patent application.

(71)	*Sökande* *Applicant (s)*	*Telefonaktiebolaget L M Ericsson (publ), Stockholm SE*
(21)	*Patentansökningsnummer* *Patent application number*	*0202058-4*
(86)	*Ingivningsdatum* *Date of filing*	*2002-07-02*

Stockholm, 2003-01-20

För Patent- och registreringsverket
For the Patent- and Registration Office

Lina Oljeqvist
Lina Oljeqvist

Avgift
Fee

Uppgjord (även faktaansvarig om annan) - *Prepared (also subject responsible if other)* Sorin Georgescu		Nr - *No.* EAB/LY-P 02:045 Uen		
Dokansv/Godk - *Doc respons/Approved*	Kontr - *Checked*	Datum - *Date* 2002-07-01	Rev	File

Voice Browsing Architecture Based on Adaptive Keyword Spotting

Inventor: Sorin Georgescu

1 TECHNICAL FIELD

Present invention is applicable in the field of voice controlled browsing and multi-modal browsing of Web content, from a dual mode voice&data Mobile Station (MS).

2 TECHNICAL BACKGROUND

2.1 THE PROBLEM AREA

Multi-modal browsing is a user friendly method that is used to access content over the Internet. When accessing content with a multi-modal browser, a user may use any of the supported input methods, or combinations thereof. Among input methods implemented so far, the most frequent ones are the key stroke method and the voice command method.

No architecture designed so far is capable of adding voice browsing functionality to an ordinary user agent running in a dual mode voice&data MS. Existing voice browsing systems are instead based on VoiceXML, a language capable of defining voice dialogs. In a VoiceXML system, the voice browsing application (speech browser) runs independently of the key-stroke browsing application. There is no synchronisation between the two browsers. Furthermore, in case the content has not been designed for both formats, i.e. HTML/XHTML and VoiceXML, there is no way to implement multi-modal browsing.

2.2 STATE OF THE ART

Voice browsing is presently implemented based on the VoiceXML paradigm. VoiceXML is a language to define voice dialogs for Internet applications accessed over the phone. In essence, output voice dialogs are carried out through audio and text-to-speech prompts, while input dialogs are carried out through touch-tone keys (DTMF) and automatic speech recognition.

A typical architecture consists of an Application Server hosting VoiceXML content, and the VoiceXML Gateway containing the speech browser (VoiceXML client) and the speech/telephony platform. The user-system interaction is done through a voice menu, from which the user can specify his selection by voice. All functionality regarding speech recognition, text-to-speech conversion, and DTMF recognition, is implemented in the speech/telephony platform, which converts to/from speech the dialogs specified in the VoiceXML page. The speech browser, based on the content interpreted on-the-fly, is the one that controls the sequence of voice dialogs. It is important to mention that by this architecture, only the voice part of the MS is used during the interaction with the user.

Read and Understood by:.. Date:.............

In above architecture, multi-modal access to Internet applications is possible only if both HTML/XHTML and VoiceXML formats are available on the Application Server. The MS used has to be a dual mode voice&data station, in order to be able to establish simultaneous voice and data sessions. Although theoretically possible, multi-modal access to dual format content requires solving a major issue, as presented in chapter 2.3.

2.3 PROBLEMS

The architectures proposed so far for voice-based applications are centred around the concept of voice dialogs usually defined in VoiceXML Therefore, only the application/content specifically designed for voice-based interaction may be accessed over the phone. The huge base of HTML/XHTML content will never be reachable, unless converted into VoiceXML.

Another problem with existing architectures is that when combining voice-based access with normal browsing, with the purpose of implementing multi-modal browsing, there is no mechanism that synchronises the two browsers, e.g. the HTML/XHTML browser running in the data part of the MS and the speech browser running in the VoiceXML Gateway. Unless a dedicated synchronisation mechanism is implemented in the Application Server, the speech browser, and the MS user agent, switching from one input method to the other during one and the same browsing session is not possible.

3 THE INVENTION

3.1 SUMMARY

The present invention overcomes the above problem by proposing an architecture and a method for the selection of speech vocabulary keywords in providing a solution to the access to an arbitrary HTML/XHTML page by means of voice commands. Multi-modal browsing is thus implemented with no need to change the original content.

3.2 DESCRIPTION

The architecture described herein is based on a speech-enabled HTTP proxy. This proxy, named in the below figure HTTP/Speech proxy, is a HTTP proxy enhanced with voice browsing functionality. It is capable of extracting keywords from browsed HTML/XHTML content as directed by predefined rules. The keywords are then emphasized in the original content so that the user will know what words to use in his speech commands when selecting a specific hyperlink.

Due to the keyword spotting nature of the voice browsing functionality, the Automatic Speech Recogniser that the proxy interfaces with can be a middle size vocabulary speech recogniser. Usually, this kind of ASR is capable of recognising continuous, speaker independent speech. Therefore, no user training is required to set-up a system with proposed architecture.

The rules used to extract the voice keywords can be grouped into:

- *Syntactic* rules – Rules like "use the subject and the predicate in a paragraph associated with a single hyperlink". Several syntactic rules prioritised with regard to the availability of keywords in the vocabulary may be used.

- *Simple* rules – Rules like "select a unique keyword in the hyperlink name, or in the paragraph associated with it". Speech commands associated to a simple rule may look like "Go to X" or "Go to the paragraph containing X".

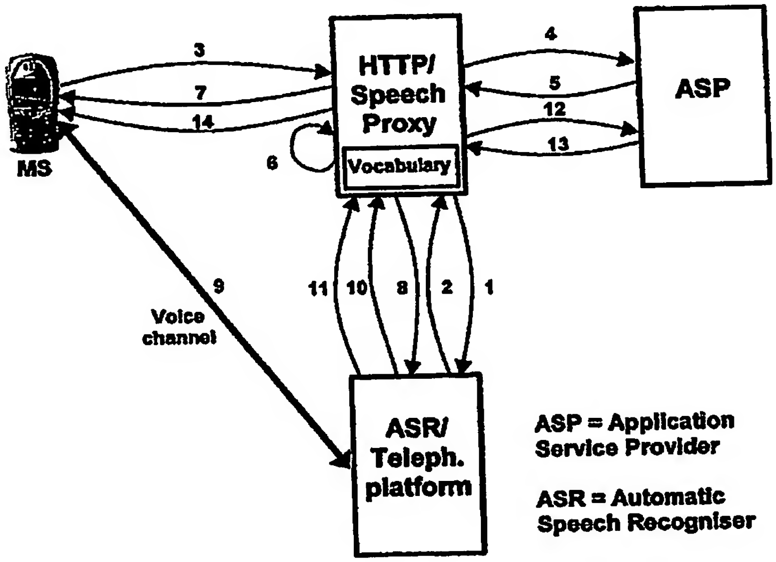
- *Numeric* rules – This refers to numbering hyperlinks in the page, or multiple hyperlinks in the same paragraph. It can be used also for option selection in menus.

During the speech interaction, the speech proxy uses speech dialogs/prompts whenever received commands are ambiguous. Standard text messages containing keywords from recognised vocabulary are forwarded to the Text To Speech (TTS) block in the Telephony Platform. The TTS block converts the messages into speech, which are then sent to the user through the voice channel. Speech dialogs may look like: "Did you select the paragraph containing keyword X?". Due to the nature of user-system interaction, the mobile stations operating with proposed architecture, need to have support for concurrent voice and data sessions.

As a general rule when implementing multi-modal browsing it is required that a synchronisation engine exists between the HTTP browser in the user agent and the speech browser, residing besides the voice browsing functionality.

This document does not address the implementation of the synchronisation engine. Instead it describes a possible way to embody this, to demonstrate that the proposed architecture can handle multi-modal access to web content in a natural way. The HTTP proxy contains a "push" mechanism that forces the user agent in the MS to refresh the content, after having fetched the page indicated through voice commands from the server. This could be based on a semaphore object (Refresh on/off) inserted in each returned page, and a script downloaded together with the page. The script forces periodic updates of the semaphore value, thus allowing the user agent to detect when a page refresh is requested by the speech proxy. Based on the semaphore's value, the script can then trigger an entire page refresh, thus downloading a fresh content.

In the following, the node interaction in the proposed architecture is described in more detail:



1. The proxy connects to the ASR/Telephony platform, specifying the application/vocabulary to connect to. This is very useful when an ASR hosts several applications implementing "flavours" of the user-system speech interface (specific sequence of voice prompts, allowed key strokes, vocabularies, etc).

2. The ASR answers back with the words contained in the vocabulary and the features supported by the Telephony platform (call-back activated, number of voice ports, etc.). An ID of the invoked application may be returned.

3. The subscriber opens a normal browsing session. In order to support voice browsing, the following information needs to be stored into the proxy's subscriber record: voice browsing On/Off, optional keyword used to trigger voice browsing, optional hyperlink name to be inserted in the accesssed Web page, which upon selection triggers the opening of the ASR-MS voice channel.

4. The proxy authenticates the user and checks whether or not voice browsing is on. The HTTP request is then forwarded to the ASP. If voice browsing is on, the proxy chooses method to open the ASR-MS voice channel based on user profile. This may be either "automatic" (steps 8 and 9), or "triggered" by user selection of some special HTTP link during browsing. In this document, reference is made only to the "automatic" establishment of the voice channel between the ASR and the MS.

5. The ASP sends back the HTML/XHTML page.

6. The proxy parses the content and analyses the paragraphs in the page using a syntactic analyser in order to find meaningful keywords. Words in hyperlink names may be selected as well as keywords. Selected keywords must be part of downloaded vocabulary, and must not lead to too close voice commands. The

keywords are then highlighted in the page through, for example, underscoring. A keyword may be present in several multi-word voice commands, provided there is sufficient discriminating information between the commands. For each browsing session the following information should to be stored: ID of the voice browsing session, subscriber's MSISDN, and selected keywords.

7. The Proxy sends modified page to the MS.

8. A voice browsing session is opened for the authenticated user with the ASR. The request should include the voice browsing session ID, the MSISDN of the user, and the application ID if provided in step 2.

9. The Telephony platform performs a call to the MS using specified MSISDN. A voice channel concurrent with the data session channel is opened between the ASR and the MS. This corresponds to the "automatic" opening of the voice channel. As mentioned earlier, the voice channel could also be switched on and off manually, through user selection of a special hyperlink. This scenario is not further dealt with in the present document.

10. After the voice channel has been opened (ie, the call is answered by the user), the ASR returns the status data to the proxy.

11. The ASR forwards to the proxy the keywords recognised in the user's voice commands. Each keyword is accompanied by its recognition probability. After analysing the keywords, the proxy tries to match them to the ones selected in step 6. In case several commands correspond, to a certain degree of confidence, to forwarded keywords, or should voice confirmation be activated, the proxy will send a text to playback to the ASR/Telephony platform. Based on the answers received from the user, the proxy will later on decide on which link to go to. (Voice prompting is not represented in the above diagram for the sake of simplicity.)

12. Using the link obtained in step 11, the proxy sends a GET request to the ASP.

13. After receiving the reply, the content is processed in a similar way as in step 6.

14. The proxy "pushes" the page to the user agent using a mechanism similar to the one described above.

Proposed voice browsing architecture and method of keyword selection solve in a natural way the issue of synchronisation between the MS user agent and the voice browser. Thus, as the speech proxy automatically extracts keywords from viewed pages, there is no need to develop a special voice format for the HTML/XHTML content. Furthermore, due to the "push" mechanism that is used to force a refresh of the content after a hyperlink has been identified using voice commands, multi-modal user input is always in sync. Regarding the vocabulary recognised by the ASR, a middle size vocabulary adjusted to the most frequently used words in the recognised language of around 2000-3000 words is probably good enough. Standard speech queries/prompts on less suggestive keywords in the paragraphs can be used in case the most suitable keywords are not part of the recognised vocabulary. VoiceXML may be a good option to define such speech queries/prompts.

3.3 **BENEFITS**

This invention paves the way for voice browsing of any web page, eliminating the need of content translation into VoiceXML or other equivalent language. It also provides the framework for multi-modal browsing, an issue that has not been addressed by state of the art systems. Because of the keyword spotting approach it allows the user to use more natural speech queries than those presently used in VoiceXML dialogs.

3.4 **BROADENING**

In case future terminals will use Voice over IP (VoIP) for the voice services, the present invention can constitute the basis for a voice browsing proxy using the data channel only.

4 **CLAIM**

An arrangement for concurrent multi-modal access of an internet page characterized in that

- the accessed page is parsed with regard to key text elements, words, and phrases, as defined by certain pre-set rules, and interpreted by a voice proxy (voice mode)
- the accessed page is browsed by means of key strokes (key stroke mode)
- the accessed page contains one set of tags alone, namely XML or the like, and thus is not in need of dual tagging ("one format fits all")

Read and Understood by:.. Date:............